The Deployment of a World Wide Web (W3) Based Medical Information System

Keith E. Willard* MD, MSEE
Jeff H. Hallgren BS
Bruce Sielaff PhD
Donald P. Connelly MD, PhD

Medical Decision Support Group, University of Minnesota Hospital and Clinic

The development of Web technologies has revolutionized information dissemination on the Internet. The University of Minnesota Hospital and Clinic's Web Clinical Information System (CIS) demonstrates the use of the Web as an infrastructure for deploying a medical information system at a fraction of the developmental cost of more traditional client server systems. This Web CIS has been deployed since December 1994. It makes available laboratory results, including a radically improved clinical microbiology reporting system, ad hoc laboratory order entry, and an embedded expert system protocol laboratory ordering system. It provides these services to any physician or patient care area with TCP (or SLIP/PPP) connection to our hospital network backbone, whether the client computer is running MS Windows, the Macintosh OS, or X-Windows. A formal evaluation of one of this systems subcomponents, the display of clinical microbiology information, demonstrated a significant savings in clinician time (43% p<.001) and substantial reduction in interpretive errors (0 vs 15 p<.01).

Introduction

Creating useful clinical information systems is a challenging task. Part of this challenge is the specification problem. It takes much effort to understand the underlying issues involved in the wide scope of areas that have to be addressed to make clinical decision support systems broadly useful. In addition, much of the utility of a decision support system may lie in individualizing the functionality of the system over substantially disparate users (*e.g.* infectious disease clinicians *and* cardiovascular surgeons).

Our previous experience with the CWS system[1,2] leads us to conclude that even with standardized open data sources such as network SQL database systems and object oriented client development environments the design, creation and deployment of custom client software still require substantial programmer resources over many months. Since successful system implementation may require multiple iterative passes through both specification and implementation, technology imposed limitations are particularly frustrating for smaller development groups. These challenges translate into substantial economic costs and represent one major reason for the slow advancement in the state of the art of medical decision support systems.

By deploying our clinical information system on top of the Web we have been able to substantially decrease the amount of effort required to deploy client services. This is accomplished by abandoning the traditional approach to custom client-side program development. We have no client-specific programming effort. Instead, all client side activities are supported by the use of a standard Web browser[3].

This has enabled us to bypass a significant portion of the development work that is customarily needed in developing a clinical system. In our experience, in comparing similar work with previous custom client development, this gain in productivity is usually at least in the range of an order of magnitude (10X).

We have developed and deployed a clinical information system that provides access via a Web browser to a variety of services including clinical laboratory results display and order entry. We have coupled a network-based inference engine to support the task of providing laboratory ordering protocols.

A pilot of this system has been operational since December of 1994 in the solid organ post-transplantation unit at the University of Minnesota Hospital and Clinic. An informal evaluation of the system has demonstrated very positive acceptance of the service by both physicians and nursing staff. A formal systematic evaluation of the system has been performed for the clinical microbiology results reporting function. It demonstrated an ~ 43% reduction in time using the new system, and a major reduction in interpretation errors.

0195-4210/95/$5.00 © 1995 AMIA, Inc.

Methods

World Wide Web

The Web technology began in the early 90's as a network-based methodology for distributing formatted documents using hypertext links as the navigation paradigm[4,5]. It evolved to include multimedia sound and graphics capabilities. The basic architecture of the Web consists of a network mounted Web server which is the network interface to all other documents or data sources on the network, and a client (the Web browser) which manages the display of the Web information documents. All user actions, such as clicks on a hypertext link, passing the results from a collection of text form fields or the setting of radio buttons, are handled by the client Web browser.

These two pieces of software cooperate via two major protocols or standards. The low level communication protocol (HTTP) manages the communication process between the client and browser that usually sits on top of a TCP network service layer. The second is HyperText Markup Language (HTML), the information/document standard for the data and forms that flow back and forth between the server and browser. HTML is an extended subset of Standard Generalized Markup Language (SGML), but efforts are underway to merge these standards[6].

While the earliest Web browsers only supported hypertext display and navigation, the current environment is considerably richer. For example, forms extensions to HTML (HTML 2.0) allow the developer to provide the user with a basic assortment of user input devices such as text/numeric fields, radio buttons, check boxes, pick lists, and action buttons. The HTML 3.0 specification includes tables, slider type controls, drag and drop file uploading, and extended sound input widgets[7]. Current extensions to the underlying network protocol include public key encryption technology that secures the network transmissions at the data packet level.

Providing an infrastructure to create useful medical decision support capability, requires far more than simply disseminating basic forms and static documents. Much of this requirement is met by the capability of the Web servers' Common Gateway Interface (CGI). This interface provides for script/program execution in lieu of a static document display. Using the CGI, the developer can create a HTML document or form on the fly in response to user interaction with the Web browser. This allows the developer to obtain data in real time from institutional sources, such as a SQL query of a network database, and encapsulate the massaged SQL response with HTML tags for dissemination back to the user.

Despite the limitations of the Web graphic user interface compared to a custom client user interface environment, we have developed user interfaces that are highly acceptable to our clinician end users. A substantial amount of time and effort previously spent on client code development and debugging has been eliminated.

The resulting application is automatically distributable over the institutional network environment. Anyone with network access to our Web servers is just a click away from our clinical information services. Remote phone access is provided with the availability of Serial Line Interface Protocol (SLIP) or Point to Point Protocol (PPP) drivers and servers. The users merely click on their Web browser. All the modem/network session establishment is initiated automatically and the user is placed into a "home page" that provides access into the Web CIS.

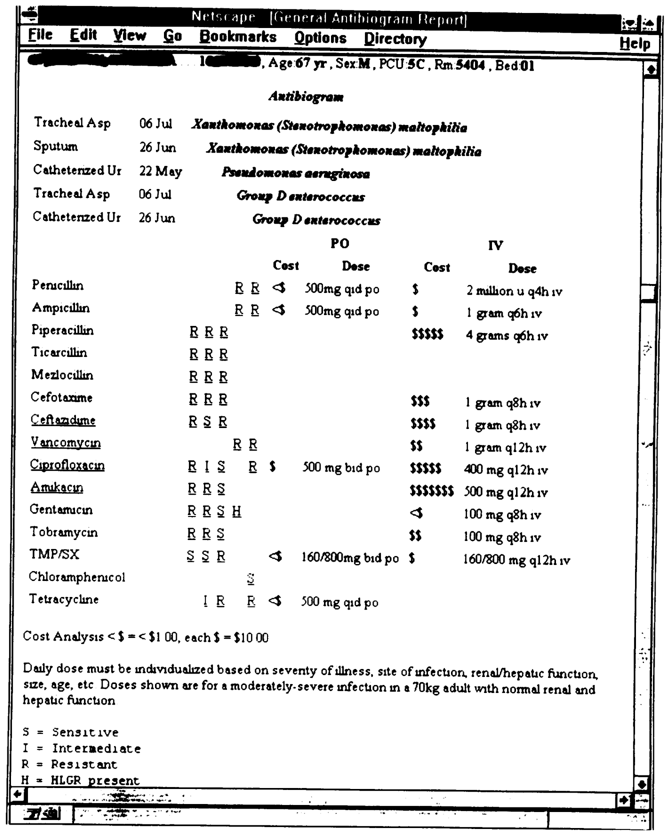

Figure 1. Antibiogram. The clinician can select any tested organism for comparison.

Development Environment

Like many other Web developers, we have found Perl to be an ideal environment for CGI scripting. It is a high level interpreted C-like language that provides excellent list processing and string manipulation facilities. It also can be conveniently extended to provide direct access to common SQL databases such as Sybase, Oracle, Informix, and DB2. Since it is interpreted, the effect of debugging changes in the Perl source can be immediately reviewed simply by selecting the browser's reload function.

The Web environment does not provide all the services needed for a basic clinical information system. We created a network security and authentication service with session-oriented security. We create user-specific server-based session context data structures to preserve user context between connection sessions. This approach obviates the usual Web practice of sending extremely long parameter strings in the Uniform Resource Locator (URL) invocation to pass on complex user context. We augmented the basic Web navigation paradigm by creating a common toolbar for each window that provides common access to each major function module.

We have great flexibility in forming effective organization displays as well as integrating on line static documents. Figure 1 demonstrates an antibiogram display that is part of the microbiology information display system. The antimicrobials with hyperlinks indicate "restricted" antibiotics and provide access to the Pharmacy and Therapeutic Committee's corresponding drug information documents.

In addition we utilize freely available packages such as gnuplot, ghostscript, and pbm+ utilities to provide network graphic services to create the postscript and gif files that we dynamically create for graphical depiction of patient data (Figure 2).

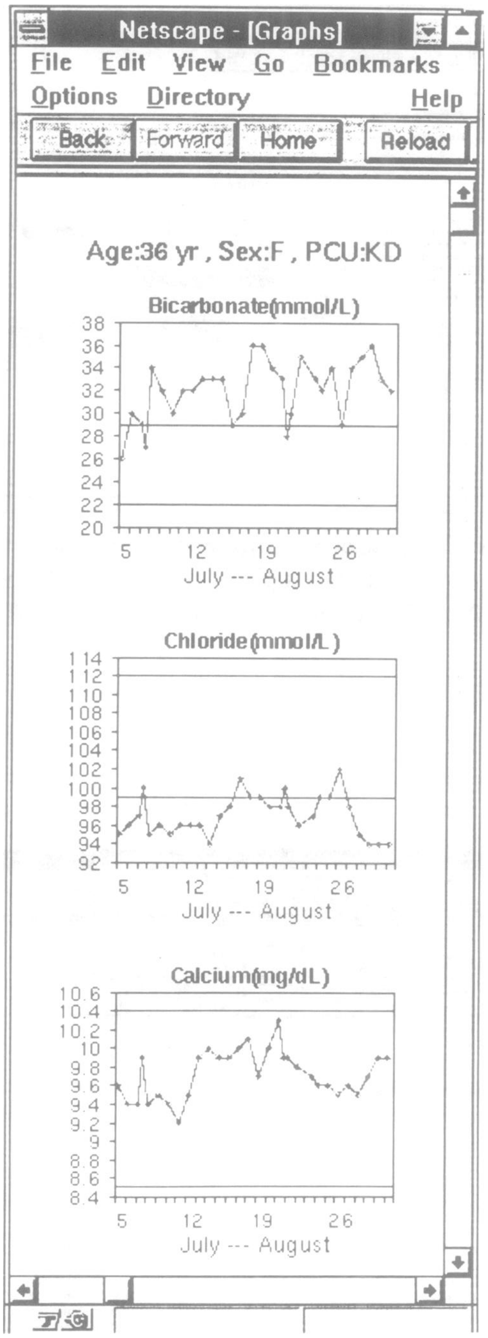

Figure 2. Graphs. The user selects the graph or graph panel and the desired time frame.

Part of our Web CIS system includes access to a network driven inference engine. The first application for this inference engine is to support automated protocol ordering. We create ordering protocols for standardized patient care pathways. For example, liver or kidney transplant patients have standard ordering patterns when the post operative clinical course is not stormy. Figure 3 demonstrates the protocol assignment window. Once a patient is assigned to a given ordering protocol all of the laboratory ordering required for that patient for their current hospitalization is automated. The results of this automated ordering are seen in Figure 4. Clinical Staff can easily cancel or reschedule any particular test and in addition perform ad hoc order entry for particular tests.

This engine was created in an object oriented dialect of Tcl, a semi lisp-like interpreted language environment. As an aside, a technical goal for this engine is to create a high level Arden syntax specification module to map to Tcl constructs. Tcl then performs the actual inference processing as well as the database accessions. This system is currently supporting more than a dozen ordering protocols and is being rapidly expanded.

All of our tools are non-proprietary and are easily moved between UNIX server environments. At the present time both Perl and Tcl have been ported to OS/2 and NT environments.

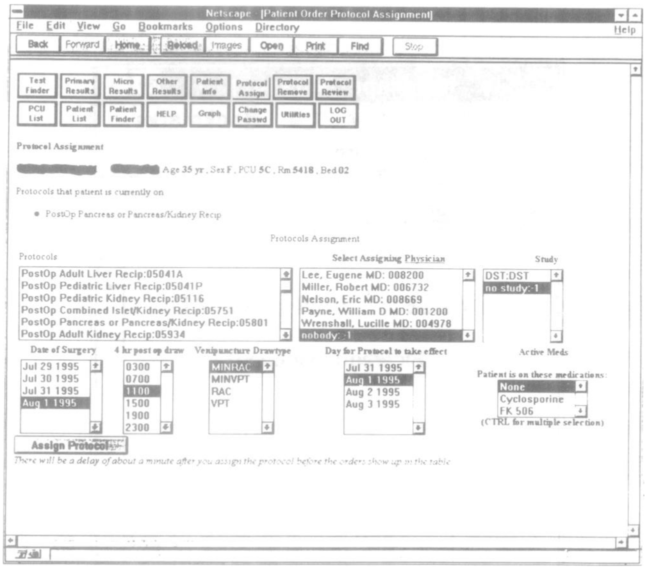

Figure 3. Protocol assignment. The clinician assigns the patient to a given care protocol along with required cognate information as a one time setup.

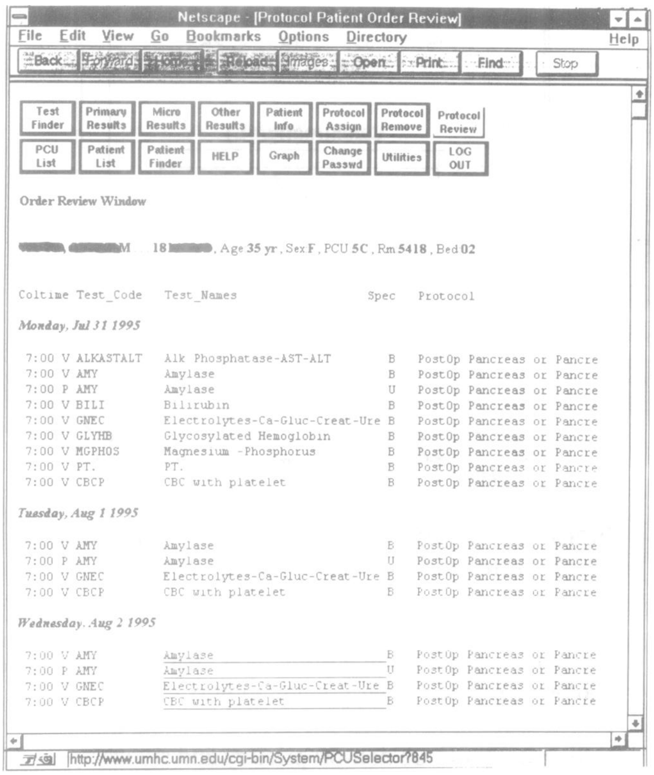

Figure 4. Protocol orders. This shows orders resulting from the automated protocol. Orders not yet ordered can be modified.

Currently deployed functionality in this system includes

- Clinical Laboratory Result Reporting
  - Primary Results window
  - Other Results modules
  - Graphical displays (either postscript or gif)
- Clinical Microbiology Display
  - Integration with on line P&T committee antimicrobial restriction and utilization documents
  - Integration with Hospital Infection Control documents and forms
- On Line Testing Catalog
  - including specimen collection requirements
- Ad Hoc Forms-based Order Entry
- Protocol Order Entry
  - using a network driven inference engine
- Hospital Policies and Procedures manuals such as
  - Infection Control, Medical Affairs newsletters

Results

In about three months we went from ground zero in terms of our knowledge of Web technology to deploying the laboratory results display component in a pilot project on our solid organ transplant service. This was accomplished with only ~2.5 programmer FTE. In contrast, the comparable part of our previous client workstation project took us about 15 months with ~5 programmer FTE.

It typically took about six months for a programmer to become facile in our earlier custom client object oriented C programming environment. By contrast it took about two weeks for them to learn enough Perl and HTML to deploy such major functionality as our Primary Results screen and the clinical microbiology display modules. For modules whose functionality was essentially preserved unaltered from our previous CWS custom client project, we saw lines of code and function count metrics drop by 90% from their previous measures.

In addition, version control is enormously simplified since there is only a single locus for the production code. We can put into place changes for all our users with a single site change rather than having to chase down to all the physician office, ward and clinic locations using the application. The greatly eases the problem of a dynamically evolving application set in a complicated institutional environment.

Only the Web server and browser are run continuously. All the rest of our code consists of a collection of small applets which only execute once per invocation. This has led to a dramatic reduction in nagging application instability problems typically caused by dangling string pointers or memory leaks that we encountered with our custom client development environment. Our application has been remarkably stable even though it is a new technology for our developers. One of our crude measures of robustness is the mean time between rebooting the application machine. This was about once per two weeks with our custom client CWS project code; it is down to once per six weeks with our Web based services.

Informally this system has been extremely well received by our clinical users but systematic formal evaluation has just begun. The first system to be evaluated was the Clinical Microbiology results reporting system. Sixteen clinicians were asked to evaluate in a structured fashion the microbiology results of two real patients. This work was done on both the preexisting terminal-based HIS system and the new Web-based CIS. The performance time of the volunteers was observed, and the accuracy of their responses was evaluated. In addition, all completed a user survey questionnaire.

The results of this evaluation demonstrated a 43% reduction in the mean time needed to complete the interpretation tasks on the Web system ($p<.001$), and a reduction in major errors (such as missing a bacteremia) from 15 to 0 ($p<.01$). There was universal agreement among participants that the system was easy to both learn and use, and that it saved substantial effort in retrieving and interpreting clinical microbiology displays.

Evaluation of the remaining functions, including an economic analysis of the protocol ordering system is underway.

Limitations

The current principle limitations are those induced by the stateless connection architecture the Web uses between browser and server. This restricts the degree of customizing possible with form data entry capabilities and required us to create an additional state layer for services, such as security, that require session oriented connection services.

Because of the Web's rapid evolution many browsers (and some servers) can behave unreliably. Our current preferred server and browser are commercially available from Netscape and have been quite stable in production versions. Our current downtime statistics are dominated by hospital network behavior or by the stability of the client operating system such as Windows or the Mac OS, rather than by our application code behavior.

Conclusions

The Web technology is not just a convenient approach to Internet connectivity. It represents a new wave in client server computing where the entire user interface application is dynamically constructed and merely interpreted on the client side, obviating the need for client side specific implementation.

We have deployed a pilot model of this technology, a clinical information system, which has demonstrated radical improvements in development productivity. This gain has translated into dramatic reductions of the development time line without trade-offs in system robustness and reliability.

While formal evaluation of the system is just beginning, preliminary results suggest that we can successfully meet a substantial portion of the functional requirements demanded by present and future clinical information systems.

References

1. **Connelly DP, Sielaff BH, and Willard KE.** A Clinician's Workstation for Improving Laboratory use. Integrated display of laboratory results. *Am J Clin Path* 1995; 104:243-52.

2. **Connelly DP, Sielaff B, Willard KE.** The Clinical Workstation as a Means of Improving Laboratory Use. *Clinica Chimica Acta.* (in press)

3. **Willard KE, Hallgren JH, Connelly DP.** W3 Based Medical Information Systems vs Custom Client Server Applications. *Proc 2nd Internat WWW Conf* 1994; 641-651.

4. **Berners-Lee T, Cailliau R, Luotonen A, Nielsen HF, Secret A.** The World Wide Web. *Communications of the ACM* 1994; 37(8) 76-82.

5. **Schatz BR, Hardin JB.** NCSA Mosaic and the World Wide Web: Global Hypermedia Protocols for the Internet. *Science* 1994; 265:895-901.

6. **Sperberg-McQueen CM, Goldstein RF.** HTML to the Max: A Manifesto for Adding SGML Intelligence to the World Wide Web. *Proc 2nd Internat WWW Conf* 1994; 189-198.

7. **Internet Engineering Task Force** HyperText Markup Language Specification Version 3.0 (IETF) 1995.